Práctica 1: Problema del Bandido de k-brazos.

Alejandro López Cuéllar, Miguel Ángel Vera Frutos, Alejandro Belda Fernández

9 de marzo de 2025

1. Introducción

El bandido de k brazos, también conocido como bandido multibrazo (*K-bandit* en inglés), hace referencia al problema en el que una persona se encuentra frente a una serie de máquinas tragaperras. De este modo, el jugador deberá seleccionar con qué máquina jugar en cada momento, ya que las recompensas ofrecidas por cada una de las máquinas vienen dadas según una distribución de probabilidad propia de cada tragaperras. Además, partiendo de la base de que el jugador no posee ninguna información previa acerca de las recompensas que ofrecen dichas máquinas.

Se establece pues como objetivos para la práctica:

- Realizar el estudio de distintos métodos empleados para abordar el problema del bandido multibrazo.
- Implementar dichos métodos estudiados.
- Realizar experimentos con cada uno de los métodos empleados para ver su funcionamiento con distintos tipos de distribuciones de probabilidad.

Así pues, a lo largo de este documento se tratarán distintos puntos relacionados con dicha práctica.
En un primer lugar, en el apartado de **Desarrollo** se especificará la definición formal del problema; tras ello, se comentará los algoritmos referentes a cada método empleado en el apartado de **Algoritmos**. Una vez comentados los algoritmos empleados, en el apartado de **Evaluación/Experimentos** se mostrarán las pruebas realizadas con cada uno de los métodos y las distintas distribuciones de probabilidad.

2. Desarrollo

2.1. Descripción

El problema del **bandido multibrazo** es un problema clásico de aprendizaje por refuerzo. Se define formalmente de la siguiente manera:

- Existen k acciones diferentes a elegir en cada instante de tiempo t.
- Elegir una de las acciones $a_t \in A$ ofrece una recompensa r_t.
- Cada uno de los brazos k posee una distribución de recompensas que es desconocida para el usuario.
- Se establece como objetivo maximizar el valor de la recompensa acumulada que se obtiene: $\sum_t r_t$.
- La presencia de distribuciones de probabilidad desconocidas para el jugador presentan un problema, de tal modo que se debe elegir entre **explorar**, probando distintos brazos para comprobar sus recompensas y **explotar**, eligiendo el brazo k que mejor valor de recompensa esperada q_a posea.

2.2. Métodos utilizados

De este modo, los métodos empleados para trabajar en el problema del bandido multibrazo han sido los siguientes:

- Método ϵ greedy
- Métodos UCB:
  - UCB1
  - UCB2
- Métodos de Ascenso de Gradiente:
  - Softmax
  - Gradiente de preferencias

2.3. Distribuciones de probabilidad

A su vez, para trabajar con las distintas distribuciones de probabilidad se han implementado funciones para generar los brazos del bandido según la distribución normal, distribución binomial y distribución bernoulli.

3. Algoritmos

3.1. Algoritmos de los métodos usados

A continuación, se realizará la explicación de los algoritmos utilizados para abarcar el problema del bandido multibrazos.

3.1.1. ϵ-Greedy

Mediante el método épsilon-greedy se pretende realizar solucionar el problema que viene dado en el método greedy, el cual está centrado en la explotación.
Mediante el valor épsilon se permitirá realizar una exploración de los brazos pese a poder no ser lo que mayor recompensa esperada posean.

Así pues, el modo de proceder es el siguiente:

- Se genera un número aleatorio entre 0 y 1.
- Se compara con $\boldsymbol{\epsilon}$, si el valor de ϵ es mayor se selecciona de forma aleatoria uno de los brazos, excluyendo el que mayor recompensa esperada posee.
- Si el valor de $\boldsymbol{\epsilon}$ es menor que el número generado, por el contrario, se seleccionará el brazo óptimo.

De esta forma, se permite cierto balance entre explotación y exploración, evitando centrarse únicamente en uno de los brazos.

Algorithm 1 Selección de brazo con ε-Greedy

Require: $k > 0$ (número de brazos), $0 \leq \varepsilon \leq 1$ (probabilidad de exploración)
Require:
Ensure: Índice del brazo seleccionado
Generar un número aleatorio $r \sim U(0, 1)$
if $r < \varepsilon$ **then**
    $chosen\_arm \leftarrow$ seleccionar un brazo al azar en $\{0, \ldots, k-1\}$
else
    $chosen\_arm \leftarrow \arg\max(values)$
end if
return $chosen\_arm$

3.1.2. UCB1

Los métodos UCB basan su forma de actuar en el **límite superior de confianza**, este valor mide el potencial de los brazos mediante la suma de dos elementos. **Q(a)**, siendo el término que representa la parte de explotación; y **u(a)** siendo el término que representa la exploración.
El hecho de que **Q(a)** esté asociado a explotación y **u(a)** a explotación viene dado por el hecho de que en la suma de estos dos valores para la obtención de **ucb(a)**, cuando uno de ellos es alto, el otro es bajo. Por tanto, si un brazo se selecciona por ser el que mayor valor ucb posee, y a su vez dicho valor ucb es grande debido a (Q(a)) es causado porque la recompensa esperada en ese brazo es alta. Sin embargo, si se selecciona por su alto valor en **u(a)** será porque se pretende realizar exploración pese a que no tenga una gran recompensa esperada.

Así pues, el método **UCB1** procede de la siguiente manera:

- En primer lugar se realiza una primera pasada por todos lo brazos, de modo que si un brazo no ha sido seleccionado se devuelve inmediatamente.
- En las siguientes selecciones, se lleva a cabo el cálculo de **t** para contabilizar el número de acciones que se han realizado.
- Se obtiene el valor **ucb1** para cada uno de los brazos.
- Finalmente se selecciona el brazo con un mayor valor **ucb1**, devolviéndolo.

Algorithm 2 Selección de brazo con UCB1

Require: $k > 0$ (número de brazos)
Ensure: Índice del brazo seleccionado
for $arm = 0$ **hasta** $k - 1$ **do**
    if $counts[arm] = 0$ **then**
        return arm
    end if
end for
$t \leftarrow \sum counts$
for $arm = 0$ **hasta** $k - 1$ **do**
    $ucb1[arm] \leftarrow values[arm] + \sqrt{\frac{2 \log t}{counts[arm]}}$
end for
$chosen\_arm \leftarrow \arg\max(ucb1)$
return $chosen\_arm$

3.1.3. UCB2

Al igual que el método UCB1, este método se basa en el **límite superior de confianza** (Upper Confident Bound). Sin embargo, a diferencia de **UCB1**, en este método se introducen las siguientes variables:

- $\boldsymbol{\alpha}$, un parámetro que sirve para ajustar el equilibrio entre explotación y exploración del método.
- k_a, variable que lleva el recuento de las épocas de una acción.
- $\boldsymbol{\tau}$, variable que determina el número de veces que se seleccionará un brazo determinado dentro de una época.

El proceso pues, es el siguiente:

- Se hace una pasada inicial por cada uno de los brazos, de tal forma que si uno de los brazos todavía no ha sido seleccionado, se devuelve de forma inmediata.
- Se comprueba si al brazo que está seleccionado le queda por ejecutar la acción alguna vez dentro de la época.
- Se obtiene el número de acciones que se han tomado, **t**.
- Para cada uno de los brazos, se calcula el valor $\boldsymbol{\tau}$
- Se calcula el valor **ucb2** para cada uno de los brazos.
- Se selecciona el brazo con un mayor valor **ucb**.
- Se calculan los valores $\tau(k_a)$ y $\tau(k_a + 1)$ y se restan. Para posteriormente obtener el número de veces que se tomará la acción del brazo seleccionado en la siguiente época.

Algorithm 3 Selección de brazo con UCB2

Require: $k > 0$ (número de brazos), $0 < \alpha < 1$ (parámetro de exploración)
Ensure: Índice del brazo seleccionado
if $brazo\_seleccionado \neq$ `None` y $rondas\_restantes > 0$ **then**
    $rondas\_restantes \leftarrow rondas\_restantes - 1$
    return $brazo\_seleccionado$
end if
for cada $brazo$ en $\{0, \ldots, k-1\}$ **do**
    if $conteos[brazo] = 0$ **then**
        $brazo\_seleccionado \leftarrow brazo$
        $\tau(k_a) \leftarrow \lceil (1+\alpha)^{k_a[brazo]} \rceil$
        $rondas\_restantes \leftarrow \tau(k_a) - k_a[brazo]$
        return $brazo$
    end if
end for
$t \leftarrow \sum conteos$
for cada $brazo$ en $\{0, \ldots, k-1\}$ **do**
    $\tau(k_a) \leftarrow \lceil (1+\alpha)^{k_a[brazo]} \rceil$
    $UCB2[brazo] \leftarrow valores[brazo] + \sqrt{\frac{(1+\alpha)\log((e \cdot t)/\tau(k_a))}{2\tau(k_a)}}$
end for
$brazo\_seleccionado \leftarrow \arg\max(UCB2)$
$\tau(k_a) \leftarrow \lceil (1+\alpha)^{k_a[brazo\_seleccionado]} \rceil$
$\tau(k_a + 1) \leftarrow \lceil (1+\alpha)^{k_a[brazo\_seleccionado]+1} \rceil$
$rondas\_restantes \leftarrow \tau(k_a + 1) - \tau(k_a)$
return $brazo\_seleccionado$

3.1.4. Softmax

El algoritmo Softmax es una estrategia de selección de acciones basada en la exploración-explotación. En lugar de

elegir siempre la acción con la recompensa esperada más alta (explotación pura) o elegir aleatoriamente (exploración pura), Softmax asigna una probabilidad a cada acción basada en su valor estimado. De esta manera, se favorecen las mejores opciones, pero aún hay una probabilidad de elegir acciones subóptimas para fomentar la exploración.

El proceso que se sigue es el siguiente:

- Se inicializan unos valores de estimación **Q(a)** que se inicializan a 0, además de inicializar una temperatura τ que indicará el nivel de exploración, cuanto más grande sea este valor las probabilidades serán más uniformes, en el caso contrario, si τ es bajo, se favorecerán los brazos con mayores valores de estimación Q.
- Se calculan los valores escalados, para ello se ajustan los valores Q dividiendolos por la temperatura τ.
- Obtenemos el valor máximo de los valores estimados para establecer una estabilidad numerica y no tener problemas al calcular exponenciales.
- Calculamos el exponencial de cada valor escalado y calculamos las probabilidades normalizando los valores exponenciales.
- Se realiza una selección al azar de un brazo dado la distribución generada anteriormente.

Algorithm 4 Selección de brazo con Softmax

Require: $k > 0$ (número de brazos), $\tau > 0$ (temperatura)
Ensure: Índice del brazo seleccionado
1: Inicializar $Q \leftarrow 0$ para cada brazo
2: Calcular los valores escalados: $valores\_escalados \leftarrow \frac{Q}{\tau}$
3: Estabilidad numérica: $max\_value \leftarrow$ máx$(valores_escalados)$
4: Calcular los valores exponenciales: $exp\_values \leftarrow \exp(valores\_escalados - max\_value)$
5: Calcular probabilidades: $p_i \leftarrow \frac{exp\_values[i]}{\sum exp\_values}$ para cada i
6: Seleccionar un brazo al azar $brazo \sim$ Distribución$(p_0, p_1, \ldots, p_{k-1})$
7: **return** $brazo$

3.1.5. Gradiente de Preferencias

El algoritmo de Gradiente de Preferencias es un método basado en gradientes donde el agente aprende una distribución de selección de acciones en lugar de simplemente estimar valores de recompensa para cada brazo. Este enfoque se diferencia de otros métodos, como ϵ-greedy o softmax, porque directamente actualiza las preferencias de las acciones utilizando la política derivada de una función softmax.

En este caso, los pasos que se siguen son los siguientes:

- Se establecen un vector de preferencias **H(a)** con valores iniciales de 0 para cada brazo. Estas preferencias no representan recompensas, sino qué tan atractiva es cada acción. Además, también establecemos una recompensa promedia que nos servirá para evaluar qué tan buenas han sido las recompensas en general. Por último, Se establece un contador de tiempo **t=0**, que indica cuántas iteraciones han pasado.
- Ahora ajustamos las preferencias de los brazos, para ello primero se calcula la función exponencial, necesitaremos normalizar el vector de preferencias restandole su valor máximo (Esto se hace como en Softmax para evitar que valores grandes de H hagan que los cálculos se desborden cuando aplicamos la función exponencial en el siguiente paso).
- Ahora calculamos las probabilidades de cada brao mediante la función softmax.
- Seleccionamos un brazo al azar siguiendo la distribución generada por las probabilidades de softmax.
- Ejecutamos la acción y recibimos la recompensa R, con este actualizamos la recompensa promedio.

Algorithm 5 Selección de brazo con Gradiente de Preferencias

Require: $k > 0$ (número de brazos), $\alpha > 0$ (tasa de aprendizaje)
Ensure: Índice del brazo seleccionado
1: Inicializar $H \leftarrow 0$ para cada brazo (preferencias iniciales)
2: Inicializar $R\_promedio \leftarrow 0$ (recompensa promedio)
3: Inicializar $t \leftarrow 0$ (contador de tiempo)
4: Estabilizar las preferencias: $H\_estable \leftarrow H -$ máx(H)
5: Calcular las probabilidades $\pi_t(a) \leftarrow \frac{\exp(H\_estable)}{\sum \exp(H\_estable)}$
6: Seleccionar un brazo $brazo \sim$ Distribución$(\pi_0, \pi_1, \ldots, \pi_{k-1})$
7: **return** $brazo$

- Por último, actualizamos las preferencias de los brazos, para cada brazo "a", si el brazo "a" ha sido seleccionado se incrementa su preferencia si su recompensa fue mejor de lo esperado (la recompensa R obtenida anteriormente es mayor que la promedia). En caso contrario, para los brazos que no fueron seleccionados, se reducen las preferencias proporcionalmente a su probabilidad.
- Repetimos todos estos pasos hasta el límite de iteraciones.

Algorithm 6 Actualización del Gradiente de Preferencias

Require: $brazo\_elegido$ (índice del brazo seleccionado), $recompensa$ (recompensa recibida)
Ensure: Actualización de las preferencias H
1: Incrementar $t \leftarrow t + 1$
2: Actualizar el promedio de recompensa: $R\_promedio \leftarrow R\_promedio + \frac{(recompensa - R\_promedio)}{t}$
3: Estabilizar las preferencias: $H\_estable \leftarrow H -$ máx(H)
4: Calcular las probabilidades actuales: $\pi_t(a) \leftarrow \frac{\exp(H\_estable)}{\sum \exp(H\_estable)}$
5: Calcular la diferencia de recompensa: $d\_recom \leftarrow recompensa - R\_promedio$
6: **for** cada $brazo$ en $\{0, \ldots, k-1\}$ **do**
7: **if** $brazo == brazo\_elegido$ **then**
8: Actualizar $H[brazo] \leftarrow H[brazo] + \boldsymbol{\alpha} \cdot d\_recom \cdot (1 - \boldsymbol{\pi}_t(brazo))$
9: **else**
10: Actualizar $H[brazo] \leftarrow H[brazo] - \boldsymbol{\alpha} \cdot d\_recom \cdot \pi_t(brazo)$
11: **end if**
12: **end for**

3.2. Implementación de las distribuciones de probabilidad

Para la implementación del conjunto de brazos con una distribución normal, la forma de proceder es la siguiente:

- Se genera un conjunto de elementos no repetidos.
- Para cada uno de los brazos, se genera un valor μ aleatorio para su media.
- Se establece una desviación estándar de 1.
- A cada uno de los brazos se asigna una tupla consistente en uno de los valores μ generados junto con el valor de la desviación estándar.

3.2.1. Distribución normal

Algorithm 7 Generación de brazos con distribución normal

Require: $k > 0$ (número de brazos), $\mu\_min < \mu\_max$ (rango de medias)
Ensure: Lista de brazos generados con distribución normal

Inicializar un conjunto vacío $\mu\_valores$
while $|\mu\_valores| < k$ **do**
  Generar un número aleatorio $\boldsymbol{\mu}$ en el rango $[\mu\_min, \mu\_max]$
  Agregar $\boldsymbol{\mu}$ a $\mu\_valores$
end while
Convertir $\mu\_valores$ a lista
Definir $\sigma \leftarrow 1{,}0$ (desviación estándar fija)
Inicializar una lista vacía *brazos*
for cada $\boldsymbol{\mu}$ en $mu\_valores$ **do**
  Crear un brazo con distribución normal $ArmNormal(\mu, \sigma)$
  Agregar el brazo a la lista *brazos*
end for
return *brazos*

3.2.2. Distribución binomial

- Se generan **k** valores dentro de [**p_min**,**p_max**], redondeando posteriormente los valores a 2 decimales.
- A cada brazo se le asigna una distribución binomial, siendo **p** es su probabilidad de éxito y **n** el número de ensayos.

Algorithm 8 Generación de brazos con distribución binomial

Require: $k > 0$ (número de brazos), $p\_min < p\_max$ (rango de probabilidades), $n > 0$ (número de ensayos)
Ensure: Lista de brazos generados con distribución binomial

Inicializar un conjunto vacío $p\_valores$
while $|p\_valores| < k$ **do**
  Generar un número aleatorio p en el rango $[p\_min, p\_max]$
  Agregar p a $p\_valores$
end while
Convertir $p\_valores$ a lista
Inicializar una lista vacía *brazos*
for cada p en $p\_valores$ **do**
  Crear un brazo con distribución binomial $ArmBinomial(p, n)$
  Agregar el brazo a la lista *brazos*
end for
return *brazos*

3.2.3. Distribución bernoulli

La distribución de bernoulli es una caso concreto de la binomial, donde el valor **n** es 1, de tal forma que para realizar esta distribución se lleva a cabo lo siguiente:

- Se genera un valor **p** distinto para cada uno de los brazos **k** dentro del rango [**p_min**,**p_max**], redondeando posteriormente a 2 decimales cada valor.
- Cada uno de los brazos se genera con una distribución Bernoulli empleando como parámetro el valor **p**.
- Se devuelve finalmente la lista de brazos generados.

Algorithm 9 Generación de brazos con distribución Bernoulli

Require: $k > 0$ (número de brazos), $p\_min < p\_max$ (rango de probabilidades)
Ensure: Lista de brazos generados con distribución Bernoulli

Inicializar un conjunto vacío $p\_valores$
while $|p\_valores| < k$ **do**
  Generar un número aleatorio p en el rango $[p\_min, p\_max]$
  Redondear p a dos decimales
  Agregar p a $p\_valores$
end while
Convertir $p\_valores$ a lista
Inicializar una lista vacía *brazos*
for cada p en $p\_valores$ **do**
  Crear un brazo con distribución Bernoulli $ArmBernoulli(p)$
  Agregar el brazo a la lista *brazos*
end for
return *brazos*

4. Evaluación/Experimentos

4.1. Epsilon-Greedy

Se realiza el experimento usando 10 brazos, cada uno de acuerdo a una distribución gaussina con desviación 1. Se realizan 500 ejecuciones de 1000 pasos cada una. Se contrastan 3 algoritmos $\boldsymbol{\epsilon}$-greedy para valores $\boldsymbol{\epsilon}$: 0.0, 0.01, y 0.1. Tres líneas de colores distintos representan diferentes valores de ϵ en el algoritmo ϵ-Greedy:

- **Azul ($\epsilon = 0$):** Representa una estrategia completamente explotadora, es decir, que siempre elige la acción que ha dado la mejor recompensa hasta ahora sin explorar nuevas opciones.
- **Naranja ($\epsilon = 0.01$):** Representa una estrategia con una pequeña probabilidad del 1 % de elegir una acción aleatoria (exploración). Esto ayudará a no explotar un mismo brazo y buscar más brazos que puedan maximizar la recompensa.
- **Verde ($\epsilon = 0.1$):** Representa una estrategia con un 10 % de probabilidad de explorar acciones aleatorias.

4.1.1. Distribución normal

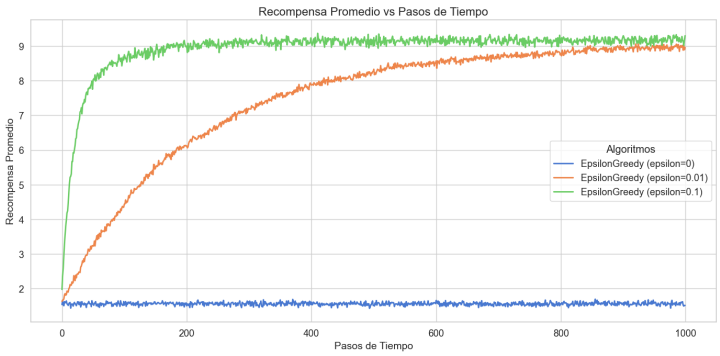

Figura 1: Recompensas promedio Epsilon Greedy en Distribución Normal

- La línea verde (ϵ=0.1) alcanza rápidamente una recompensa promedio alta, lo que indica que la estrategia con mayor exploración aprende más rápido qué brazos del bandit son óptimos.
- La línea naranja (ϵ=0.01) también muestra un crecimiento, pero más lento en comparación con ϵ=0.1.
- La línea azul (ϵ=0) se mantiene en un nivel bajo de recompensa, lo que sugiere que no logra encontrar el mejor brazo porque no explora nuevas opciones.

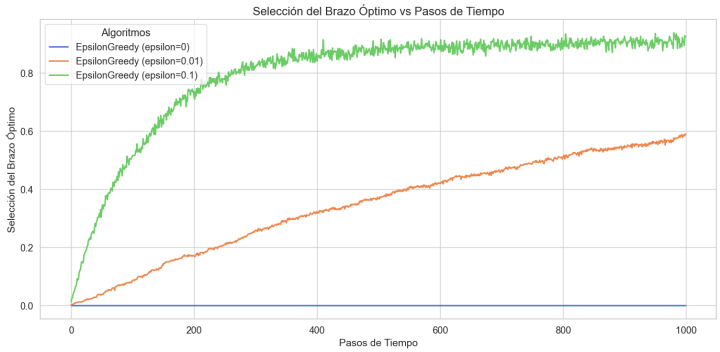

Figura 2: Selección del brazo óptimo Epsilon Greedy en Distribución Normal

- La línea verde (ϵ=0.1) alcanza rápidamente el brazo óptimo lo que indica que la estrategia con mayor exploración aprende más rápido cuál es el brazo óptimo y una vez conocido como sabe que es el que mejor recompensa promedio le puede dar, pues comienza a explotar esa opción.
- La línea naranja (ϵ=0.01) también muestra un crecimiento, pero más lento en comparación con ϵ=0.1. Por lo que tardaría algo más en encontrar el brazo óptimo, pero también lo encontraría y lo comenzaría a explotar aunque más tardamente; haciendo así que tenga un menor número de tiradas de dicho brazo.
- La línea azul (ϵ=0) al no tener exploración no logra encontrar el mejor brazo porque no explora nuevas opciones, por lo que nunca tira del brazo optimo y se dedica a explotar el brazo elegido inicialmente.

Cabe destacar que los resultados obtenidos mediante las distribuciones Binomial y Bernoulli para el caso de ϵ-greedy son prácticamente idénticos a los obtenidos mediante la distribución Normal. La única diferencia apreciable es que los picos mostrados es las gráficas de Binomial y Bernoulli son más acentuados debido al rango de valores que ofrecen como recompensa.

4.2. UCB

Se realiza el experimento usando 10 brazos, cada uno de acuerdo a una distribución gaussina con desviación 1. Se realizan 500 ejecuciones de 1000 pasos cada una. Se contrastan 3 algoritmos UCB, siendo uno de ellos UCB1 y los otros 2 UCB2 con valores alpha de 0.1 y 0.01.

Tres líneas de colores distintos representan diferentes valores de alpha en el algoritmo UCB:

- **Azul (UCB1):** Representa una estrategia que varía entre exploración y explotación según Q(a) + u(a). Donde u(a) = $\sqrt{\frac{2 \ln t}{N_t(a)}}$, siendo **t** el número total de veces que han seleccionado los brazos, y N_t(a) el número de veces que se ha seleccionado un brazo en concreto.
- **Naranja (UCB2; alpha = 0.1):** Representa una estrategia que varía entre exploración y explotación según Q(a) + u(a). Donde u(a) = $\sqrt{\frac{(1+\alpha)\ln\left(\frac{t}{\tau(k_a)}\right)}{2\tau(k_a)}}$. Con alpha a 0.1, permite que haya cierta exploración para encontrar el brazo óptimo.
- **Verde (UCB2; alpha = 0.01):** Representa una estrategia que varía entre exploración y explotación según **Q(a) + u(a). Donde u(a) = $\sqrt{\frac{(1+\alpha)\ln\left(\frac{t}{\tau(k_a)}\right)}{2\tau(k_a)}}$. Con alpha a 0.01, lo que aumenta aún más la exploración con objetivo de encontrar el brazo óptimo.

4.2.1. Distribución normal

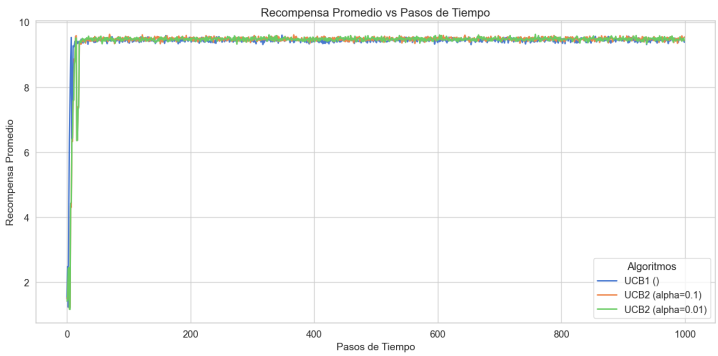

Figura 3: Recompensas promedio UCB en Distribución Normal

- La línea azul (UCB1) explora en primer lugar todos los brazos y en muy pocos pasos alcanza el brazo óptimo.
- La línea naranja (UCB2; $\boldsymbol{\alpha}$ = 0.1) muestra también un inicio explorando todos los brazos, y llega al brazo óptimo en unos pocos pasos más que UCB1.
- La línea verde (UCB2; $\boldsymbol{\alpha}$ = 0.01) al igual que los otros dos casos, llega al brazo óptimo en pocos pasos, además, se ve como debido al valor de $\boldsymbol{\alpha}$ de 0.01 lleva a cabo una mayor exploración que lo que se aprecia en UCB y UCB2

con α=0.1.

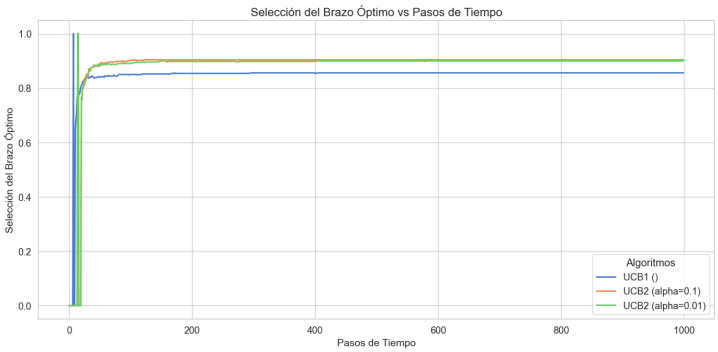

Figura 4: Selección del brazo óptimo UCB en Distribución Normal

- La línea azul (UCB1) en un primer lugar pasa por el brazo óptimo ya que se debe pasar por todos los brazos, tras ello, no tarda en alcanzar el brazo óptimo, aunque se aprecia que el número de veces que se selecciona es inferior a los de UCB2.
- La línea naranja (UCB2; $\alpha = 0.1$) realiza de igual forma una pasada por todos los brazos de forma que se observa que al principio se selecciona el brazo óptimo, tras ello, en pocos pasos se alcanza el brazo óptimo que se selecciona en mayor medida que UCB1.
- La línea verde (UCB2; $\alpha = 0.01$) refleja que en una primera pasada por todos los brazos, el óptimo es seleccionado, una vez hecho esto, en unos pocos pasos más que los 2 casos anteriores, debido a una mayor exploración por α=0.01, se alcanza el brazo óptimo.

4.2.2. Distribución binomial

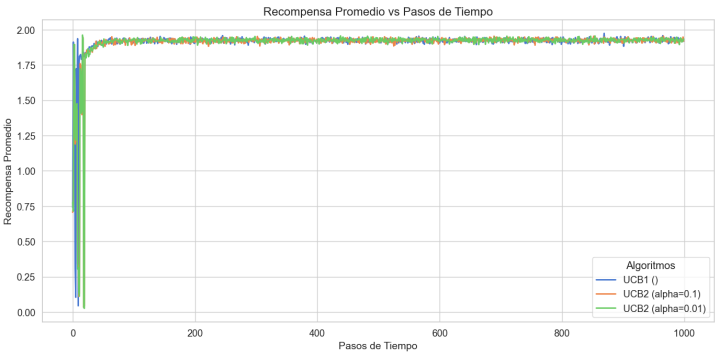

Figura 5: Recompensas promedio UCB en Distribución Binomial

- La línea azul (UCB1) tras unos pocos pasos, y tras pasar por todos los brazos en una primera vuelta, alcanza el brazo óptimo. De modo que comienza explorando los brazos y una vez da con el óptimo comienza a explotar.
- La línea naranja (UCB2; $\alpha = 0.1$) al igual que UCB1, hace una pasada por todos lo brazos y tras ello, en pocos pasos alcanza el óptimo, comenzando a explotar.
- La línea verde (UCB2; $\alpha = 0.01$) es el que mayor componente de exploración posee, debido al valor de α bajo. Por lo que tarda unos pocos pasos más en alcanzar el óptimo, debido a ese caracter exploratorio.

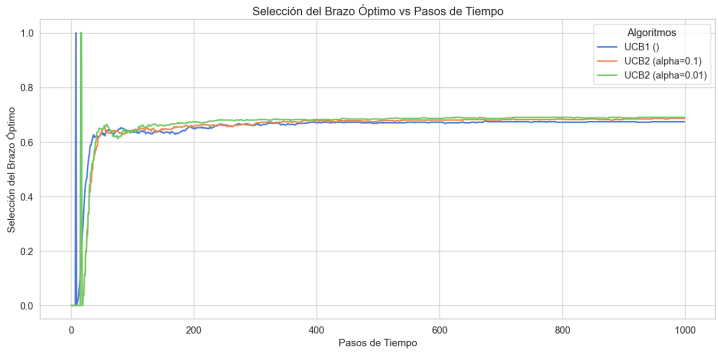

Figura 6: Selección del brazo óptimo UCB en Distribución Binomial

- La línea azul (UCB1) se observa como es la primera que alcanza el brazo óptimo, sin embargo, es la que en menor porcentaje lo selecciona, aunque sea por un margen pequeño.
- La línea naranja (UCB2; $\alpha = 0.1$) tras realizar mayor exploración, y tardar unos pocos pasos más, alcanza el brazo óptimo, es notable en esta ocasión, como el brazo óptimo es seleccionado en menor medida que con la otra distribución.
- La línea verde (UCB2; $\alpha = 0.01$) de forma similar a la línea naranja, lleva a cabo mayor exploración que UCB1, tras ello, alcanza el brazo óptimo, el cual lo selecciona en mayor medida que lo apreciable en la línea azul, y de forma ligeramente superior también a lo mostrado en la línea naranja.

Lo más destacable que notamos es que respecto a la distribución normal el brazo óptimo de promedio es elegido en una menor frecuencia, esto debido a la exsitencia de un brazo subóptimo que posee una recompensa esperada muy similar a la del brazo óptimo.

4.2.3. Distribución bernouilli

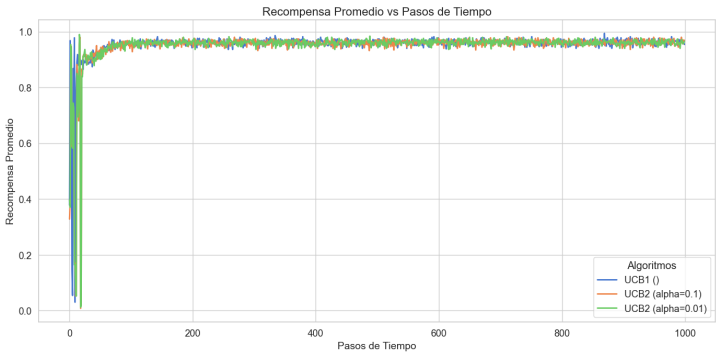

Figura 7: Recompensas promedio UCB en Distribución Bernoulli

- La línea azul (UCB1) siguiendo las tendencias anteriores, tras unos pocos pasos alcanza el brazo óptimo, donde comienza a explotar.

- La línea naranja (UCB2; $\boldsymbol{\alpha} = 0.1$) de igual forma, comienza probando con todos los brazos y tras ello, en pocos pasos es capaz de alcanzar el brazo óptimo.
- La línea verde (UCB2; $\boldsymbol{\alpha} = 0.01$) alcanza también el brazo óptimo en unos pocos pasos. A diferencia de lo que ocurría con otras distribuciones de probabilidad, en esta ocasión, los tres alcanzan el brazo óptimo prácticamente con el mismo número de pasos.

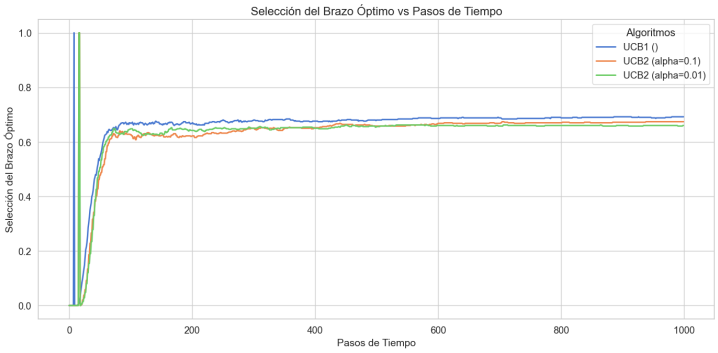

Figura 8: Selección del brazo óptimo UCB en Distribución Bernoulli

- La línea azul (UCB1) comienza con la exploración y alrededor de los 70 steps alcanza el brazo óptimo. En una primera instancia el brazo óptimo es seleccionado por realizar una primera pasada por cada uno de los brazos. Vemos aún así, que el porcentaje de selección del brazo óptimo es similar a lo que ocurría con la distribución de probabilidad binomial.
- La línea naranja (UCB2; $\boldsymbol{\alpha} = 0.1$) explora por los brazos en una primera ronda, y tras ello, en pocos pasos, de igual forma que UCB1 alcanza el brazo óptimo, aunque se puede observar que lo selecciona un número de veces lígeramente inferior.
- La línea verde (UCB2; $\boldsymbol{\alpha} = 0.01$) alcanza rápidamente el brazo óptimo lo que indica que la estrategia con mayor exploración aprende más rápido cuál es el brazo óptimo y una vez conocido como sabe que es el que mejor recompensa promedio le puede dar, pues comienza a explotar esa opción. Aunque menos que en experimentos previos.

4.3. Ascenso del Gradiente

- **Azul (Softmax, $\tau = 0.1$):** Representa una estrategia altamente explotadora, lo que significa que rara vez explora otras opciones y tiende a quedarse con las primeras elecciones que considera buenas.
- **Naranja (Softmax, $\tau = 0.5$):** Introduce algo más de exploración en la selección del brazo, pero aún así no logra una alta tasa de selección del brazo óptimo en comparación con otras estrategias.
- **Verde (Softmax, $\tau = 1$):** Tiene un nivel de exploración alto, lo que se traduce en un rendimiento creciente, pero con una selección del brazo óptimo inferior a otras estrategias basadas en Gradiente de Preferencias.
- **Rojo (Gradiente de Preferencias, alpha = 0.1):** Modifica las preferencias de los brazos con un ajuste moderado en cada actualización.
- **Lila (Gradiente de Preferencias, alpha = 0.5):** Aumenta la tasa de actualización, permitiendo cambios más rápidos en las preferencias.
- **Marrón (Gradiente de Preferencias, alpha = 1):** Aplica una tasa de aprendizaje aún mayor, ajustando las preferencias con mayor intensidad en respuesta a las recompensas recibidas.

4.3.1. Distribución normal

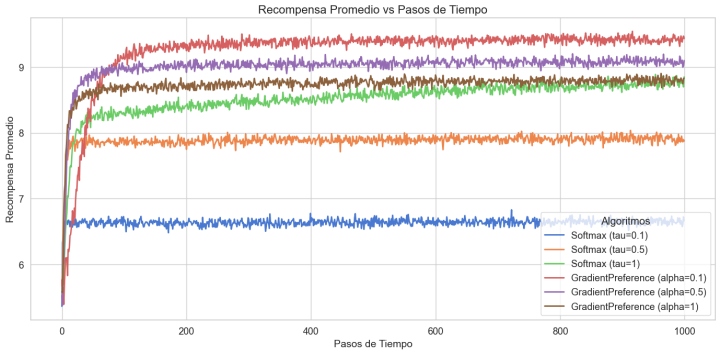

Figura 9: Recompensas promedio Ascenso del Gradiente en Distribución Normal

- La línea **azul (Softmax, $\tau = 0.1$)** mantiene una baja tasa de recompensa promedio, lo que indica que no selecciona el brazo óptimo por su tendencia a la explotación.
- La línea **naranja (Softmax, $\tau = 0.5$)** muestra que su tasa de recompensa promedio es mayor que con '$\tau = 0.1$', pero sigue siendo limitada debido a una exploración moderada.
- La línea **verde (Softmax, $\tau = 1$)** se mantiene en un nivel alto de recompensa, pero sigue manteniendo por debajo de los algoritmos de **Gradiente de Preferencias**
- La línea **marrón (Gradiente de Preferencias, alpha = 1)** mejora progresivamente la recompensa promedio, pero de manera menos eficiente que valores menores de α. Su alta tasa de actualización hace que las preferencias cambien de forma más brusca, lo que puede generar inestabilidad en la selección del brazo óptimo.
- La línea **lila (Gradiente de Preferencias, alpha = 0.5)** presenta una mejora significativa con respecto a '$\alpha = 1$', estabilizándose alrededor de 0.5. Su menor tasa de actualización permite que la estrategia se adapte de manera más efectiva a las recompensas recibidas.
- La línea **roja (Gradiente de Preferencias, alpha = 0.1)** a pesar de su baja tasa de aprendizaje, se beneficia de la estabilidad en sus actualizaciones, lo que permite una convergencia consistente hacia el brazo óptimo, alcanzando casi un 85 % de selección tras suficientes pasos de tiempo.

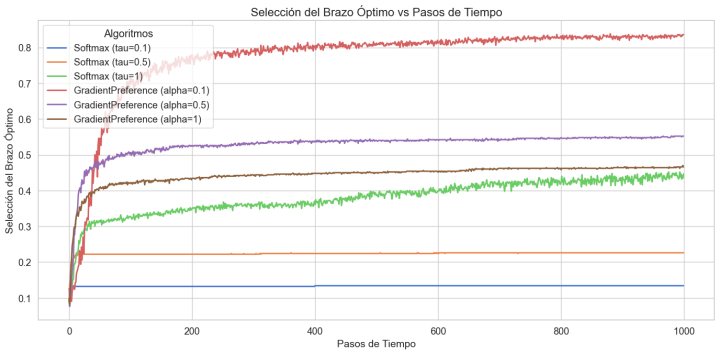

Figura 10: Selección del brazo óptimo Ascenso del Gradiente en Distribución Normal

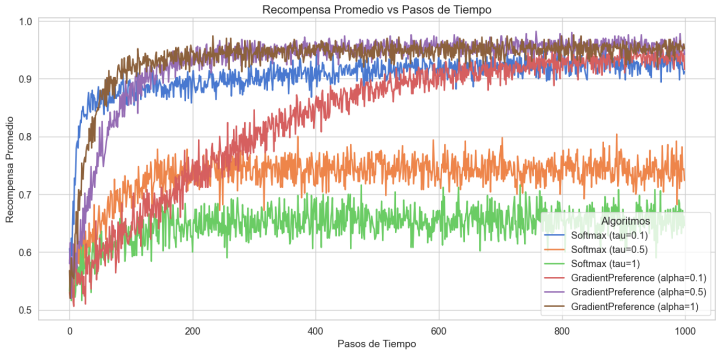

Figura 11: Recompensas promedio Ascenso del Gradiente en Distribución Normal

- La línea **azul (Softmax, $\tau = 0.1$)** mantiene en un valor cercano a 0, lo que indica que rara vez selecciona el brazo óptimo. Esto sugiere que la estrategia es altamente explotadora y se queda atrapada en elecciones subóptimas iniciales sin explorar nuevas opciones.
- La línea **naranja (Softmax, $\tau = 0.5$)** mantiene una selección del brazo óptimo baja, estabilizándose en torno a 0.2. Aunque introduce algo de exploración, sigue sin ser suficiente para identificar el brazo óptimo con frecuencia.
- La línea **verde (Softmax, $\tau = 1$)** explora más en comparación con los otros valores de "τ", pero aún así no logra identificar consistentemente el brazo óptimo, estabilizándose alrededor de 0.3.
- La línea **marrón (Gradiente de Preferencias, $\alpha = 1$)** va mejorando la selección del brazo óptimo a lo largo del tiempo y se estabiliza en torno a 0.4. Esto sugiere que, aunque hay una adaptación progresiva, la tasa de aprendizaje es baja y tarda en encontrar la mejor opción.
- La línea **lila (Gradiente de Preferencias, $\alpha = 0.5$)** presenta una mejora significativa con respecto a '$\alpha = 0.1$', estabilizándose alrededor de 0.5. Su mayor tasa de actualización permite que la estrategia se adapte más rápido a las recompensas recibidas.
- La línea **roja (Gradiente de Preferencias, $\alpha = 0.1$)** es la estrategia que más selecciona el brazo óptimo, superando casi un 85 % de selección tras suficientes pasos de tiempo. Su alta tasa de aprendizaje permite ajustar rápidamente las preferencias hacia el brazo óptimo, logrando un desempeño superior al resto.

4.3.2. Distribución binomial

- La línea **verde (Softmax, $\tau = 1$)** se mantiene en un nivel alto de recompensa (ya que aunque es la peor, estamos en una distribución donde la recompensa promedio máxima sería de 2) pero es el primer en converger en una recompesa promedio de 1.4, lo que indica que explora más que los otros valores de 'τ', pero aún así no logra identificar consistentemente el brazo óptimo.
- La línea **naranja (Softmax, $\tau = 0.5$)** y la linea **azul (Softmax, $\tau = 0.1$)** tienen un comportamiento similar, pero la primera tiene una convergencia un poco más lenta que la segunda(mientras que la naranja para obtener su valor máximo en los primeros pasos de tiempo, el azul empieza a converger en el paso numero 100 o 150). Aún así, siguen teniendo una recompensa promedio inferior a el algoritmo Gradiente de Preferencias.
- Las líneas correspondientes a Gradiente de Preferencias como la **roja (Gradiente de Preferencias, alpha = 0.1)**, la **lila (Gradiente de Preferencias, alpha = 0.5)** y la **marrón (Gradiente de Preferencias, alpha = 1)** también tienen un comportamiento similar, pero la roja tiene una convergencia más lenta que las otras dos (se estabiliza en el paso número 400) y la lila tiene una convergencia un poquito más lenta que la marrón. Aún así, todas convergen a una recompensa promedio de 1.8, lo que indica que explora más que los valores de 'τ' de Softmax. También nos indica en este caso que un valor más alto de alpha nos permite adaptarnos más rápido y converger antes a la recompensa promedio máxima.

Algo destacable es que las recompensas promedio son más bajas que para la distribución anterior, pero esto puede deberse simplemente al cómo esta distribución genera las recompensas haciendo que tengan un valor más bajo.

Figura 12: Selección del brazo óptimo Ascenso del Gradiente en Distribución Normal

- La línea **azul (Softmax, $\tau = 0.1$)** mantiene en un valor cercano a 0.2, lo que indica que rara vez selecciona el brazo óptimo. Esto sugiere que la estrategia es altamente explotadora y se queda atrapada en elecciones subóptimas iniciales sin explorar nuevas opciones.
- La línea **naranja (Softmax, $\tau = 0.5$)** mejora su selección del brazo óptimo en comparación con '$\tau = 0.1$', pero sigue sin ser suficiente para identificar el brazo óptimo con frecuencia.

- La línea **verde (Softmax, $\tau = 1$)** en este caso tiene un comportamiento peculiar respecto a distribución normal, ya que empeora el comportamiento respesta a '$\tau = 0.5$', puede ser que debido al comportamiento tan explorador, no llegue a centrarse en el brazo óptimo.
- La línea **marrón (Gradiente de Preferencias, $\alpha = 1$)** es el primero en converger de los algoritmos de Gradiente de Preferencias, pero su selección del brazo óptimo es inferior a los otros valores de 'α'.
- La línea **lila (Gradiente de Preferencias, $\alpha = 0.5$)** presenta una convergencia más lenta que '$\alpha = 1$', pero su selección del brazo óptimo es superior.
- La línea **roja (Gradiente de Preferencias, $\alpha = 0.1$)** es la estrategia que más selecciona el brazo óptimo, superando un 60 % de selección tras suficientes pasos de tiempo. Aunque como se ve en la tabla, tarda bastante en converger en ese resultado, estando en una frecuencia más baja que sus homólogos casi hasta el paso 400.

Lo más destacable que notamos es que pareciese que no hubiese una relación directa entre la recompensa promedio y la selección del brazo óptimo, ya que en general todos los algoritmos tenían una recompensa promedio elevada (como mínimo un valor de 1.4 de 2.0) pero se selecciona el brazo óptimo en una frecuencia muy baja. Esto se debe (como se verá más adelante) a que uno de los brazos subóptimos tiene una recompensa promedio muy parecida a la óptima, haciendo que aunque no se elija el brazo óptimo, la recompensa promedio sea alta.

4.3.3. Distribución bernouilli

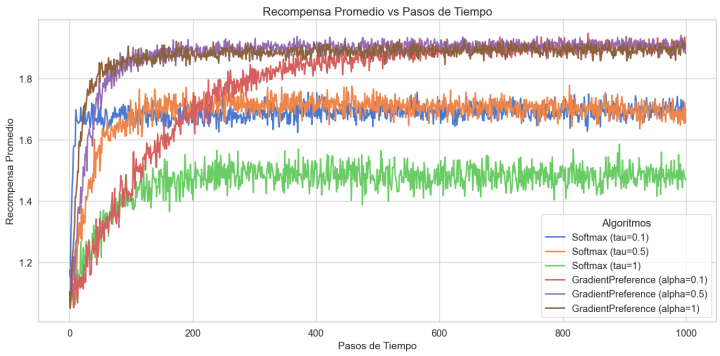

Figura 13: Recompensas promedio Ascenso del Gradiente en Distribución Normal

- La línea **verde (Softmax, $\tau = 1$)** mantiene una baja tasa de recompensa promedio, lo que indica que la alta exploración impide la selección eficiente del mejor brazo.
- La línea **naranja (Softmax, $\tau = 0.5$)** logra una recompensa mayor que "$\tau = 1$", pero sigue limitada por su exploración moderada.
- La línea **azul (Softmax, $\tau = 0.1$)** es la mejor dentro de Softmax, alcanzando una recompensa alta, pero con menor capacidad de adaptación.
- La línea **marrón (Gradiente de Preferencias, alpha = 1)** mejora progresivamente la recompensa promedio, aunque su alta tasa de actualización puede introducir fluctuaciones.
- La línea **lila (Gradiente de Preferencias, alpha = 0.5)** encuentra un buen equilibrio, estabilizándose en una recompensa cercana a 0.95, aunque realiza una convergencia más lenta que la linea marrón y azul.
- La línea **roja (Gradiente de Preferencias, alpha = 0.1)** tiene una convergencia más lenta pero estable, alcanzando cerca de 0.95 de recompensa promedio.

Figura 14: Selección del brazo óptimo Ascenso del Gradiente en Distribución Normal

- La línea **verde (Softmax, $\tau = 1$)** mantiene en un valor cercano a 0.2, lo que indica que rara vez selecciona el brazo óptimo. Esto sugiere que la estrategia es altamente exploradora no llega a centrarse en el brazo óptimo.
- La línea **naranja (Softmax, $\tau = 0.5$)** mejora su selección del brazo óptimo en comparación con '$\tau = 1$', pero sigue sin ser suficiente para identificar el brazo óptimo con frecuencia.
- La línea **azul (Softmax, $\tau = 0.1$)** en este caso tiene un comportamiento mejor que los otros valores de 'τ', pero sigue sin ser suficiente para identificar el brazo óptimo con frecuencia (30 %).
- La línea **marrón (Gradiente de Preferencias, $\alpha = 1$)** es el primero en converger de los algoritmos de Gradiente de Preferencias, pero su selección del brazo óptimo es inferior a los otros valores de 'α'. Ocurriendo algo parecido a la explicación de los brazos de distribución binomial.
- La línea **lila (Gradiente de Preferencias, $\alpha = 0.5$)** presenta una convergencia más lenta que '$\alpha = 1$', pero su selección del brazo óptimo es superior.
- La línea **roja (Gradiente de Preferencias, $\alpha = 0.1$)** es la estrategia, junto a la linea lila, que más selecciona el brazo óptimo, superando un 60 % de selección tras suficientes pasos de tiempo. Anteriormente, la convergencia en distribuciones binomiales era más rápida (por el paso 400), en este caso, este algoritmo supera a los otros pero mucho más tarde, en el paso 800.

5. Conclusiones

Se pueden extraer las siguientes conclusiones:

- En cuanto a ϵ-greedy se puede apreciar como los resultados entre las distintas distribuciones han sido muy parecidos. El punto a destacar es que según se ha ido aumentando el valor del ϵ, los resultados de las recompensas promedio obtenidas mejoran, ya que aumenta la exploración que realiza el método, permitiendo así alcanzar el brazo óptimo con una mayor facilidad.

- Respecto a los métodos UCB, se puede apreciar que los resultados obtenidos tanto en UCB1 como en UCB2 con los distintos valores de alpha, en esta ocasión han sido bastante similares, balanceando entre exploración y explotación para alcanzar el brazo óptimo. Lo más destacable es cómo en el caso de binomial y bernoulli el brazo óptimo no era seleccionado tantas veces como en la distribución normal debido a la presencia de un brazo subóptimo con una recompensa similar a la del brazo óptimo.
- En lo métodos de ascenso de gradiente, podemos ver que de manera general, los métodos de Gradiente de preferencias funcionan de forma adecuada en distintos escenarios. Por otro lado, en el caso del método Softmax aparecen diferencias algo más notorias. Mientras que para la distribución normal los algoritmos más exploradores obtienen unos mejores resultados, cuando se está trabajando con las distribuciones binomial y bernoulli, aquellos métodos en los que se ha empleado una estrategia más basada en la explotación han sido los que mejores recompensas promedio han obtenido.
  Cabe comentar, que de forma similar a lo que ocurre con UCB, cuando se trabaja con distribuciones binomial o bernoulli, se aprecia una menor selección del brazo óptimo debido a la presencia de un brazo subóptimo con una recompensa promedio bastante similar a la de dicho brazo óptimo.

6. Repositorio

El repositorio en el que se realiza el trabajo es el siguiente:https://github.com/DeMiKe16/k_brazos_VLB.git